WHAT IS CLAIMED IS:

1. A display system, comprising:

a display;

a processor coupled between the display and an application program interface (API) and adapted to execute code within the API during runtime of an application program, wherein execution of said code by the processor generates an image upon the display, such that a look and feel of the image displayed using the first API is the same across diverse operating systems, and wherein the API lacks functionality provided by a second API within a second display system;

software components adapted for incorporation into an API; and

a third API, resulting from the incorporation of the software components into the first API and capable of providing at least some of the functionality present in the second API and absent in the first API, and retaining the look and feel consistency of the first API.

2. The display system as recited in claim 1, wherein the image generated comprises pixels presented upon the display via the graphical user interface associated with the application program.

3. The display system as recited in claim 1, wherein the image contains representations of buttons, list boxes and slide bars on which a pointer device can be directed by a user.

4. The display system as recited in claim 1, wherein the application program runs under a standard computer operating system, such as Windows, Unix or OS/2.

5. The display system as recited in claim 1, wherein the application program is written in Java programming language.

6. The display system as recited in claim 5, wherein the first API comprises Java Swing.

7. The display system as recited in claim 5, wherein the functionality lacked by the first API comprises support for Unicode font encoding and font searching capability.

8. The display system as recited in claim 7, wherein the functionality lacked by the first API further comprises the use of an advanced font rasterizer for the generation of high quality text.

9. The display system as recited in claim 8, wherein the functionality lacked by the first API further comprises enhanced text support, including popup menus with cut and paste editing capability, and undo/redo editing.

10. The display system as recited in claim 9, wherein the functionality lacked by the first API further comprises consistently proper menu bar behavior, independent of the operating system under which the application program is running.

11. The system as recited in claim 1, wherein the image presents a consistent look and feel upon the display independent of the operating system under which the application program is running.

12. A method for displaying an image, comprising:

running an application program upon a computer, wherein the application program is coupled to a first API adapted for the display of images, such that a look and feel of the images displayed using the first API is consistent across diverse operating systems, and wherein the first API lacks functionality provided by a second API;

replacing the first API with a third API, created by incorporating into the first API software components that confer at least some of the functionality present in the second API and absent in the first, and wherein the third API retains the look and feel consistency of the first API; and

re-running the application program.

13. The method as recited in claim 12, wherein said displaying comprises presenting pixels upon the display via a graphical user interface associated with the application program.

14. The method as recited in claim 12, wherein the displaying the image comprises presenting representations of buttons, list boxes and slide bars upon the display on which a pointer device can be directed by a user.

15. The method as recited in claim 12, wherein the application program runs under a standard computer operating system, such as Windows, Unix or OS/2

16. The method as recited in claim 12, wherein the application program is written in Java programming language.

17. The method as recited in claim 16, wherein the first API comprises Java Swing.

18. The method as recited in claim 17, wherein the functionality lacked by the first API comprises support for Unicode font encoding and font searching capability.

19. The method as recited in claim 18, wherein the functionality lacked by the first API further comprises the use of an advanced font rasterizer for the generation of high quality text.

20. The method as recited in claim 19, wherein the functionality lacked by the first API further comprises enhanced text support, including popup menus with cut and paste editing capability, and undo/redo editing.

21. The method as recited in claim 21, wherein the functionality lacked by the first API further comprises consistently proper menu bar behavior, independent of the operating system under which the application program is running.

22. A computer-readable storage device, comprising:

a windows-based operating system; and

an application program adapted for executing code of a software component,

from a first API adapted to generate a first image independent of executing code within the operating system;

from a second API adapted to generate a second image, and providing functionality not present in the first API; and

from a third API adapted to generate a third image independent of executing code within the operating system, wherein the third API is formed by incorporating additional software components into the first API, and providing at least some of the functionality present in the second API and absent from the first.

23. The computer-readable storage device as recited in claim 22, wherein the application program is written in Java programming language.

24. The computer-readable storage device as recited in claim 22, wherein the first API is Java Swing.